

WHAT IS CLAIMED IS:

1. A multiple processor computer system comprising:
a plurality of processors;
a shared resource;
a main memory, at least a portion thereof comprising a control structure for controlling a lock on said shared resource; and
a crossbar structure for controlling access among the processors to the shared resource, the crossbar structure comprising, for each processor, a corresponding storage location, one of the plurality of processors writing to a storage location corresponding to the one processor, an address of the lock control structure associated with said shared resource to acquire the lock thereto, the crossbar structure, on behalf of the one processor, performing memory operations on the lock control structure at the address specified in the corresponding storage location in order to acquire the lock on behalf of the one processor.
2. The multiple processor computer system of claim 1 wherein the crossbar structure further comprises a second storage location corresponding to a respective processor in the multi-processor system, one of the plurality of processors writing to the corresponding second storage location an address of the lock control structure associated with said shared resource to release the lock thereto, the crossbar structure, on behalf of the processor, performing memory operations on the lock control structure at the address specified in the corresponding second storage location in order to release the lock on behalf of the processor.
3. The multiple processor computer system of claim 2 wherein the crossbar structure further comprises a third storage location corresponding to a respective processor in the multi-processor system for storing a current interrupt priority level associated with the processor wherein the current interrupt priority level represents the level of interrupt that the processor will service while spinning on a lock.
4. The multiple processor computer system of claim 3 wherein the crossbar structure further comprises a fourth storage location corresponding to a respective

processor in the multi-processor system for storing a future interrupt priority level associated with the processor wherein the future interrupt priority level represents the level of interrupt that the processor will operate at upon receiving a lock.

5. The multiple processor computer system of claim 1 wherein the crossbar structure further comprises a queue for determining which processor is granted a lock.

6. The multiple processor computer system of claim 5 wherein the queue comprises a first data field for containing a processor identification, a second data field for containing a lock request time, and a third data field for containing a priority of a task requesting a lock.

7. The multiple processor computer system of claim 6 wherein the queue determines which processor is granted a lock based on the lock request time and the priority of the task requesting the lock.

8. The multiple processor computer system of claim 1 further comprising a multi-processor module.

9. The multiple processor computer system of claim 8 wherein the multi-processor module comprises at least two of the plurality of processors.

10. The multiple processor computer system of claim 1 wherein the shared resource comprises at least a portion of memory.

11. The multiple processor computer system of claim 1 wherein the shared resource comprises an I/O device.

12. The multiple processor computer system of claim 1 wherein the shared resource comprises a register.

for each processor, a corresponding fourth storage location for storing a future interrupt priority level associated with the processor wherein the future interrupt priority level represents the level of interrupt that the processor will operate upon receiving a lock

18. The crossbar structure of claim 17 wherein the queue comprises a first data field for containing a processor identification, a second data field for containing a lock request time, and a third data field for containing a priority of a task requesting a lock.

19. The crossbar structure of claim 18 wherein the queue determines which processor is granted a lock based on the lock request time and the priority of the task requesting the lock.

20. A method for acquiring a lock to a shared resource in a multiprocessor computer system, the multiprocessor computer system comprising a plurality of processors, a main memory, and a crossbar structure comprising, for each processor, a first storage location for storing a memory address of a control structure associated with said shared resource to be locked, the method comprising:

writing, by one of said processors, the memory address of said control structure to said first storage location corresponding to said one processor when said one processor needs to acquire a lock on said shared resource; and

performing, by said crossbar structure, on behalf of said one processor, a memory operation on the control structure at the memory address stored in said corresponding first storage location in an attempt to acquire a lock to said shared resource; and

returning a status of the memory operation to said one processor.

21. The method of claim 20 wherein performing a memory operation comprises performing a test and set operation.

22. The method of claim 20 further comprising storing a current interrupt priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

23. The method of claim 20 further comprising storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

24. The method of claim 23 further comprising determining a processor from the plurality of processors to be granted a lock based on a queue.

25. The method of claim 24 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

26. The method of claim 23 wherein storing the future interrupt priority level further comprises copying a value contained in a current interrupt priority register to a future interrupt priority register prior to writing, by one of said processors, the memory address.

27. The method of claim 26 further comprising copying a value contained in a future interrupt priority register to a current interrupt priority register.

2025 RELEASE UNDER E.O. 14176

28. A method for use in a multi-processor computer system having a crossbar structure that connects a plurality of processors to at least one shared resource, the method comprising:

receiving at the crossbar from one of said processors, a memory address of a lock control structure associated with said shared resource when the processor needs to acquire a lock thereto; and

performing at the crossbar structure, on behalf of said one processor, a memory operation on the lock control structure at the memory address received from said one processor in order to acquire the lock on behalf of said one processor.

29. The method of claim 28 further comprising storing the memory address received from said one processor in a respective storage location within the crossbar structure.

30. The method of claim 28 further comprising determining a status of the lock control structure at the memory address received from said one processor.

31. The method of claim 30 further comprising sending to said one processor the determined status of the lock control structure.

32. The method of claim 28 further comprising performing at the crossbar structure memory operations on the lock control structure at the memory address received from said one processor, continually, until an unlocked status is received.

33. The method of claim 28 wherein performing a memory operation comprises performing a test and set operation.

34. The method of claim 28 further comprising receiving a current interrupt priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

35. The method of claim 28 further comprising storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

36. The method of claim 28 further comprising determining a processor from the plurality of processors to be granted a lock based on a queue.

37. The method of claim 36 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding to the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

109861705 : 050901

38. A computer-readable medium having instructions stored thereon for use in a multi-processor computer system having a crossbar structure that connects a plurality of processors to at least one shared resource, the instructions, when executed on the crossbar structure, causing the crossbar structure to perform the following:

receiving at the crossbar from one of said processors, a memory address of a lock control structure associated with said shared resource when the processor needs to acquire a lock thereto; and

performing at the crossbar structure, on behalf of said one processor, a memory operation on the lock control structure at the memory address received from said one processor in order to acquire the lock on behalf of said one processor.

39. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform storing the memory address received from said one processor in a respective storage location within the crossbar structure.

40. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform determining a status of the lock control structure at the memory address received from said one processor.

41. The computer-readable medium of claim 40 wherein the instructions further cause the crossbar to perform sending to said one processor the determined status of the lock control structure.

42. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform performing at the crossbar structure memory operations on the lock control structure at the memory address received from said one processor, continually, until an unlocked status is received.

43. The computer-readable medium of claim 38 wherein performing a memory operation comprises performing a test and set operation.

44. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform receiving a current interrupt priority level associated with said one processor wherein the current interrupt priority level represents the level of interrupt that said one processor will service while spinning on a lock.

45. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform storing a future interrupt priority level associated with said one processor wherein the future interrupt priority level represents the level of interrupt that said one processor will service upon receiving a lock.

46. The computer-readable medium of claim 38 wherein the instructions further cause the crossbar to perform determining a processor from the plurality of processors to be granted a lock based on a queue.

47. The computer-readable medium of claim 46 wherein the determining a processor further comprises:

reading at least one priority and a processor identification corresponding to each priority, from the queue;

selecting the highest priority of the read at least one priority;

determining a processor based on the processor identification corresponding the selected highest priority; and

granting the lock to the determined processor corresponding to the selected highest priority.

2025-03-27 10:00:00